ELSEVIER

Theoretical Computer Science 155 (1996) 321–347

Theoretical Computer Science

Reconstructing convex polyominoes from horizontal and vertical projections

Elena Barcucci[a,*], Alberto Del Lungo[a], Maurice Nivat[b], Renzo Pinzani[a]

[a]*Dipartimento di Sistemi e Informatica, Università di Firenze, Via Lombroso 6/17, Firenze, Italy*
[b]*LITP Institut Blaise Pascal, Université Paris 7 "Denis Diderot", 2 Place Jussieu, Paris Cedex 05, France*

Received November 1994
Communicated by E. Goles

Abstract

Reconstructing discrete bidimensional sets from their projections is involved in many different problems of computer-aided tomography, pattern recognition, image processing and data compression. In this paper, we examine the problem of reconstructing a discrete bidimensional set S satisfying some convexity conditions from its two orthogonal projections (H, V). We develop an algorithm that starts out from (H, V) and reconstructs set S, when S is a convex polyomino, in polynomial time. At the same time, we show that determining the existence of a row-convex (column-convex) polyomino or set with connected rows (columns) having assigned orthogonal projections (H, V) is an NP-complete problem. Moreover, by using the algorithm to reconstruct convex polyominoes from their two orthogonal projections we prove that the *numerical matching with target sums* problem can be solved in polynomial time if its sequences are unimodal.

1. Introduction

A *cell* is a unitary square $[i, i+1] \times [j, j+1]$, in which $i, j \in \mathbb{N}_0$. Let S be a finite set of cells. The ith *row projection* and the jth *column projection* of S are the number of cells in the ith row and in the jth column of S, respectively. In this paper, we examine an aspect of the reconstruction of objects from their projections: that of establishing the existence of an S set of cells, in which the ith row projection and the jth column projection are equal to h_i and v_j, respectively, and $H = (h_1, h_2, \ldots, h_m) \in \mathbb{N}^m$ and $V = (v_1, v_2, \ldots, v_n) \in \mathbb{N}^n$ are two assigned vectors. Determining the existence of an S set having assigned projections (H, V) means establishing

* Corresponding author. E-mail: pire@ingfii.ing.unifi.it.

SSDI 0304-3975(94)00293-2

the existence of solutions to the following system in $n + m$ equations and in $n + m$ binary variables $s_{i,j}$:

$$\sum_{j=1}^{n} s_{i,j} = h_i, \quad 1 \leqslant i \leqslant m,$$
$$\sum_{i=1}^{m} s_{i,j} = v_i, \quad 1 \leqslant j \leqslant n, \tag{1.1}$$

A set S of cells can be represented by a matrix of 0 and 1, that is, a binary pattern. First Ryser [23], and subsequently Chang [5] and Wang [29] studied the problem of proving the existence of solutions to system (1.1) and therefore to binary patterns S having assigned projections (H, V) and they showed that the decision problem can be solved in $O(nm)$ time. These authors also developed some algorithms that reconstruct S starting out from (H, V). Reconstructing a binary pattern from its projections is of primary importance in medical diagnostics (computer-aided tomography), pattern recognition, image processing and data compression [6–8, 17, 19, 20, 22, 25, 26]. The main problem met with in the reconstruction is "ambiguity" involved because, in some cases, a great many sets have the same projections (H, V). For example, if $H = (h_1, h_2, \ldots, h_m) = (1, 1, \ldots, 1)$ and $V = (v_1, v_2, \ldots, v_n) = (1, 1, \ldots, 1)$, there are $n!$ different sets having these projections. The following two methods have been developed to diminish all ambiguity.

(i) More than two projections are assigned (the algorithms that can use multiple projections are described in [7, 19, 24]).

(ii) Some of the properties of the set to be reconstructed are given "a priori" (for example, convexity, connection, symmetries) and the algorithms take advantage of this further information to reconstruct the set (see [6, 11, 20–22]). For example Chang and Chow [6] defined an algorithm that reconstructs binary patterns which are convex and symmetrical with respect to the two orthogonal axis; Kuba's heuristic algorithm [20] reconstructs patterns having connected rows and columns.

As far as the latter method is concerned, some properties imposed on the sets entirely eliminate all ambiguity (see [11]), while other properties only partially reduce it. It is shown in [12] that there is an exponential number of convex sets having the same projections. Therefore, when convexity conditions are imposed on the sets, ambiguity is reduced but not eliminated. We also noted that, in some cases, ambiguity reduction does not facilitate the set's reconstruction. As a matter of fact, when certain convexity conditions are imposed, it becomes difficult both to reconstruct the set by means of efficient algorithms and to establish their existence. For this reason, it is important to solve the decision problem regarding the existence of a set S having assigned projections (H, V) subject to some convexity constraints. Such a solution allows us to know which of the constraints allow the problem to be solved in polynomial time and which of them make the problem NP-complete.

In this paper, the decision problem is studied on connected finite sets of adjacent cells lying two by two along a side. These sets are the well-known combinatorial

objects called *polyominoes* (see [13, 15, 18]) and they are related to many different problems, such as:

Tiling: We examine the possibility of tiling finite figures of $\Pi = \mathbf{Z} + \mathbf{Z}$ [3,9,16] and the whole plane Π [2,4] by means of a finite set of assigned polyominoes.

Enumerating: We examine the possibility of enumerating polyominoes according to their various parameters [10, 28]. Such problems are also studied by physicists because they serve as models for many physical phenomena [27].

In Section 3, we show that the decision problem regarding the class of convex polyominoes can be solved in polynomial time. In Section 4, we prove that the problem regarding the classes of row-convex (column-convex) polyominoes and of sets having connected rows (columns) is NP-complete. We also use the algorithm that reconstructs convex polyominoes to show that the *numerical matching with target sums* problem (NMTS, see [14]) can be solved in polynomial time if its sequences are unimodal.

2. Preliminaries

Let S be a finite set of cells. A *column* (*row*) of S is the intersection of S with infinite vertical strip $[i, i+1] \times \mathbb{R}$ (horizontal $\mathbb{R} \times [i, i+1]$), in which $i \in \mathbb{N}_0$.

Notations. We say that a set S of cells satisfies properties **p**, **v** and **h** if

p: S is a polyomino,

v: every column of S is a connected set,

h: every row of S is a connected set.

A set S belongs to class (**x**) ($S \in (\mathbf{x})$) iff it satisfies the property **x**.

We now introduce the following definitions:

S is a convex polyomino if $S \in (\mathbf{p}, \mathbf{v}, \mathbf{h})$,

S is a column-convex polyomino if $S \in (\mathbf{p}, \mathbf{v})$,

S is a row-convex polyomino if $S \in (\mathbf{p}, \mathbf{h})$.

We introduce the notion of satisfiability because it allows us to define the problem of reconstructing a cell set S from its projections.

Definition 2.1. Let $H = (h_1, h_2, \ldots, h_m) \in \mathbb{N}^m$ and $V = (v_1, v_2, \ldots, v_n) \in \mathbb{N}^n$. The pair (H, V) is said to be satisfiable in class (**x**) if there is at least one set $S \in (\mathbf{x})$ such that S's ith row projection and jth column projection are equal to h_i and v_j, respectively, for $i = 1, 2, \ldots, n$ and $j = 1, 2, \ldots, m$. We also say that S satisfies (H, V) in (**x**).

We can now define the following problem:

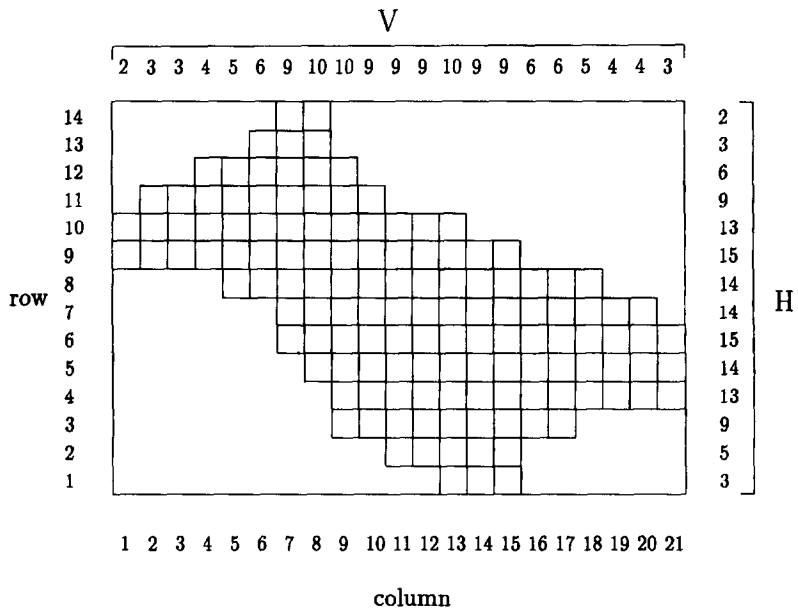

Fig. 1. A convex polyomino that satisfies (H, V).

Reconstructing a set S from its projections (RSP)

Instance: Two vectors $H = (h_1, h_2, \ldots, h_m) \in \mathbb{N}^m$, $V = (v_1, v_2, \ldots, v_n) \in \mathbb{N}^n$ and a class (**x**) of cell sets.

Question: Is the pair (H, V) satisfiable in (**x**)?

Example 1. Let H and V be as follows:

$$H = (3, 5, 9, 13, 14, 15, 14, 14, 15, 13, 9, 6, 3, 2),$$

$$V = (2, 3, 3, 4, 5, 6, 9, 10, 10, 9, 9, 9, 10, 9, 9, 6, 6, 5, 4, 4, 3).$$

The convex polyomino in Fig. 1 satisfies (H, V) and therefore (H, V) is satisfiable in (**p**, **v**, **h**).

The following proposition directly follows from the above definition.

Proposition 2.2. *Let $H = (h_1, h_2, \ldots, h_m) \in \mathbb{N}^m$ and $V = (v_1, v_2, \ldots, v_n) \in \mathbb{N}^n$. If (H, V) is satisfiable in* (**x**), *then*:

(i) $\forall i \in [1..m]\ 1 \leqslant h_i \leqslant n$,

(ii) $\forall j \in [1..n]\ 1 \leqslant v_i \leqslant m$,

(iii) $\sum_{j=1}^{m} h_j = \sum_{i=1}^{n} v_i$.

Consequently, if a set of cells $S \in (\mathbf{x})$ satisfies a pair (H, V) having $H \in \mathbb{N}^m$ and $V \in \mathbb{N}^n$, then S is contained in a rectangle R of size $n \times m$ (see Fig. 1).

3. The RSP problem on (p, v, h)

In this section, we define an algorithm that establishes the existence of a convex polyomino satisfying a pair of assigned vectors (H, V) in polynomial time. We then prove that RSP problem on **(p, v, h)** is solved in polynomial time.

Let us take two vectors $H \in \mathbb{N}^m$ and $V \in \mathbb{N}^n$ and a convex polyomino Λ that satisfies (H, V). From Proposition 2.2 it follows that Λ is contained in a rectangle R of size $n \times m$. Let $[S, S']$ ($[E, E']$, $[N, N']$, $[W, W']$) be the intersection of Λ's boundary on the lower (right, upper, left) side of R. Segment $[S, S']$ is the base of a set made up of h_1 consecutive columns of Λ, called Λ's *foot*, denoted as P_S (see Fig. 2). In the same way, we define Λ's other three feet P_E, P_N and P_W by referring to intersections $[E, E']$, $[N, N']$ and $[W, W']$. Given a pair of vectors (H, V), we make some attempts to reconstruct the polyomino to find out if there is at least one convex polyomino Λ that satisfies it. We show that, if all our attempts fail (that is, we are not able to reconstruct Λ), then (H, V) is not satisfiable in **(p, v, h)**.

Our algorithm's first step is to check to see if (H, V) satisfies Proposition 2.2's conditions. If so, we go on to construct Λ's feet. Λ's foot P_S is made up of h_1 columns and so it can have $n - h_1 + 1$ possible positions. Let us examine the ith position of P_S, in which the P_S's first column is R's ith column (see Fig. 2). We denote this foot as $P_S(i)$ and find that it is made up of the columns going from the ith to the $(i + h_1 - 1)$th column, whose lengths are equal to $v_i, v_{i+1}, \ldots, v_{i+h_1-1}$, respectively. The number of possible positions for feet P_N, P_E and P_W is equal to $n - h_m + 1$, $m - v_1 + 1$ and $m - v_n + 1$, and, therefore, the number of possible positions for the four feet is $(n - h_1 + 1)(n - h_m + 1)(m - v_1 + 1)(m - v_n + 1) \leqslant n^2 m^2$. For each of possible

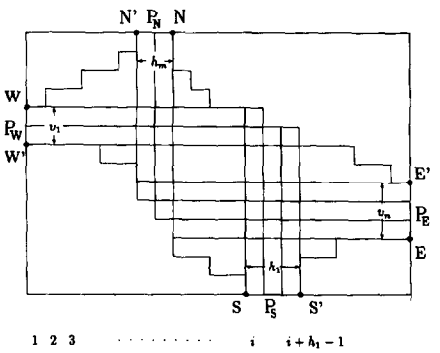

Fig. 2. Feet of a convex polyomino.

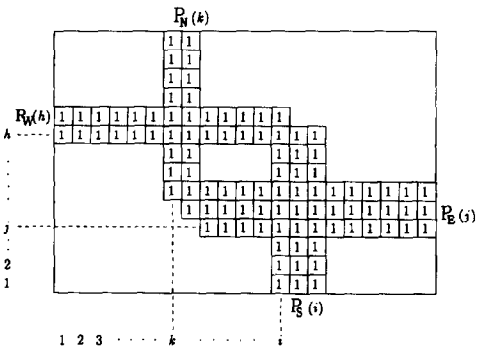

Fig. 3. Position $Q(i, j, k, h) = (P_S(i), P_E(j), P_N(k), P_W(h))$ of the four feet.

positions, we attempt to construct a convex polyomino Λ satisfying (H, V). Assuming that (i, j, h, k) is one of these positions, we denote $Q(i, j, k, h)$ as the set made up of feet $P_S(i)$, $P_E(j)$, $P_N(k)$, $P_W(h)$ (see Fig. 3).

We assume that a convex polyomino Λ exists that satisfies (H, V) such that $Q(i, j, k, h) \subset \Lambda$. We call *kernel* any set α of cells such that $Q(i, j, k, h) \subseteq \alpha \subseteq \Lambda$, and we call *shell* any set β of cells such that $\Lambda \subseteq \beta \subseteq R$. Consequently, the shell contains Λ, while the kernel is contained in Λ. Assuming that $\alpha := Q(i, j, k, h)$ and $\beta := R$, the basic idea of the algorithm that reconstructs Λ by starting out from $Q(i, j, k, h)$ consists in reducing the shell and expanding the kernel by means of *filling operations* that take advantage of the convexity constraint and vectors H and V. The shell is reduced by eliminating the cells not belonging to Λ from β. Vice versa, the kernel is expanded by putting the cells belonging to Λ into α. If polyomino Λ exists, the reconstruction algorithm stops when $\alpha = \Lambda = \beta$. If Λ does not exist, the reconstruction fails, that is, the filling operations produce a kernel α and a shell β such that $\alpha \not\subset \beta$. In this case there is no convex polyomino having its feet in position (i, j, k, h) that satisfies (H, V) and so the algorithm is performed starting from another position (i_1, j_1, k_1, h_1).

We label α's cells "1" and the ones not belonging to β "0". Since $Q(i, j, k, h) \subseteq \alpha$, the cells belonging to the feet are labelled "1". We now define the filling operations acting on R's rows and columns that expand the kernel and reduce the shell. Since the operations are performed on the rows and the columns, we introduce the following notations:

R^i, α^i, β^i indicate R's ith row, and the cells of α and β belonging to R's ith row (that is, $\alpha^i = \alpha \cap R^i$ and $\beta^i = \beta \cap R^i$), respectively.

R_j, α_j, β_j indicate R's jth column, and the cells of α and β belonging to R's jth column (that is, $\alpha_j = \alpha \cap R_j$ and $\beta_j = \beta \cap R_j$), respectively.

$c(i, j)$ indicates the cell belonging to R's ith row and jth column.

$l(\alpha^i) \stackrel{\text{def}}{=} \min\{j \in [1..n]: c(i, j) \in \alpha^i\}$ and $r(\alpha^i) \stackrel{\text{def}}{=} \max\{j \in [1..n]: c(i, j) \in \alpha^i\}$,

$d(\alpha_j) \stackrel{\text{def}}{=} \min\{i \in [1..m]: c(i, j) \in \alpha_j\}$ and $u(\alpha_j) \stackrel{\text{def}}{=} \max\{i \in [1..m]: c(i, j) \in \alpha_j\}$.

It is worth noting that $c(i, l(\alpha^i))$ $(c(i, r(\alpha^i)))$ is the leftmost (rightmost) cell in α^i, while $c(d(\alpha_j), j)$ $(c(u(\alpha_j), j))$ is the lowest (uppermost) cell in α_j.

Kernel expansion

Connecting operation $\oplus$

On row R^i:

(a_1) if $\alpha^i = \emptyset$, then $\oplus(\alpha^i) = \alpha^i$;

(a_2) if $\alpha^i \neq \emptyset$, then $\oplus(\alpha^i) = \{c(i,j)\colon l(\alpha^i) \leqslant j \leqslant r(\alpha^i)\}$.

On column R_j:

(b_1) if $\alpha_j = \emptyset$, then $\oplus(\alpha_j) = \alpha_j$;

(b_2) if $\alpha_j \neq \emptyset$, then $\oplus(\alpha_j) = \{c(i,j)\colon d(\alpha_j) \leqslant i \leqslant u(\alpha_j)\}$.

Coherence operation $\otimes$

On row R^i:

(a_1) if β^i is disconnected, then $\otimes(\alpha^i) = \alpha^i$;

(a_2) if β^i is connected (that is, if there are $j_1, j_2 \in [1..n]$ such that $\beta^i = \{c(i,j)\colon j_1 \leqslant j \leqslant j_2\}$) and $\alpha^i = \emptyset$, then $\otimes(\alpha^i) = \{c(i,j)\colon j_2 - h_i + 1 \leqslant j \leqslant j_1 + h_i - 1\}$;

(a_3) if β^i is connected (that is, $\beta^i = \{c(i,j)\colon j_1 \leqslant j \leqslant j_2\}$) and $\alpha^i \neq \emptyset$, then $\otimes(\alpha^i) = \alpha^i \cup \{c(i,j)\colon l(\alpha^i) \leqslant j \leqslant j_1 + h_i - 1$ or $j_2 - h_i + 1 \leqslant j \leqslant r(\alpha^i)\}$.

On column R_j:

(b_1) if β_j is disconnected, then $\otimes(\alpha_j) = \alpha_j$;

(b_2) if β_j is connected (that is, if there are $i_1, i_2 \in [1..m]$ such that $\beta_j = \{c(i,j)\colon i_1 \leqslant i \leqslant i_2\}$) and $\alpha_j = \emptyset$, then $\otimes(\alpha_j) = \{c(i,j)\colon i_2 - v_j + 1 \leqslant i \leqslant i_1 + v_j - 1\}$;

(b_3) if β_j is connected (that is, $\beta_j = \{c(i,j)\colon i_1 \leqslant i \leqslant i_2\}$) and $\alpha_j \neq \emptyset$, then $\otimes(\alpha_j) = \alpha_j \cup \{c(i,j)\colon d(\alpha_j) \leqslant i \leqslant i_1 + v_j - 1$ or $i_2 - v_j + 1 \leqslant i \leqslant u(\alpha_j)\}$.

The cells put into the kernel by both operations are labelled "1".

Shell reduction

Connecting operation $\ominus$

On row R^i:

(a_1) if $\alpha^i = \emptyset$, then $\ominus(\beta^i) = \beta^i$;

(a_2) if $\alpha^i \neq \emptyset$ and $c(i,k) \notin \beta^i$, then, if $k < l(\alpha^i)$, we get $\ominus(\beta^i) = \beta^i - \{c(i,j)\colon j \leqslant k\}$, while if $k > r(\alpha^i)$, we get $\ominus(\beta^i) = \beta^i - \{c(i,j)\colon j \geqslant k\}$.

On column R_j:

(b_1) if $\alpha_j = \emptyset$, then $\ominus(\beta_j) = \beta_j$;

(b_2) if $\alpha_j \neq \emptyset$ and $c(k,j) \notin \beta_j$, then, if $k < d(\alpha_j)$, we get $\ominus(\beta_j) = \beta_j - \{c(i,j)\colon i \leqslant k\}$, while if $k > u(\alpha_j)$, we get $\ominus(\beta_j) = \beta_j - \{c(i,j)\colon i \geqslant k\}$.

Coherence operation $\odot$

On row R^i:

(a_1) if $\alpha^i = \emptyset$, then $\odot(\beta^i) = \beta^i$;

(a_2) if $\alpha^i \neq \emptyset$, then $\odot(\beta^i) = \beta^i - \{c(i,j)\colon j \leqslant r(\alpha^i) - h_i \text{ or } j \geqslant l(\alpha^i) + h_i\}$.

On column R_j:

(b_1) if $\alpha_j = \emptyset$, then $\odot(\beta_j) = \beta_j$;

(b_2) if $\alpha_j \neq \emptyset$, then $\odot(\beta_j) = \beta_j - \{c(i,j)\colon i \leqslant u(\alpha_j) - v_i \text{ or } i \geqslant l(\alpha_j) + v_j\}$.

The cells eliminated from the shell by both operations are labelled with "0".

Example 2. Let us assume that R^i is made up of 16 cells and has $\beta^i = \{c(i,j)\colon 2 \leqslant j \leqslant 12 \text{ and } 14 \leqslant j \leqslant 15\}$ and $\alpha^i = \{c(i,3), c(i,6)\}$. We denote the cells in $\beta - \alpha$, that is, the unlabelled cells, as λ and obtain

$$R^i = 0\,\lambda\,1\,\lambda\,\lambda\,1\,\lambda\,\lambda\,\lambda\,\lambda\,\lambda\,\lambda\,0\,\lambda\,\lambda\,0$$

Since $l(\alpha^i) = 3$ and $r(\alpha^i) = 6$, by performing operation $\oplus$ on R^i, we obtain $\oplus(\alpha^i) = \{c(i,j)\colon 3 \leqslant j \leqslant 6\}$,

$$R^i = 0\,\lambda\,1\,1\,1\,1\,\lambda\,\lambda\,\lambda\,\lambda\,\lambda\,\lambda\,0\,\lambda\,\lambda\,0$$

Since $c(i,13) \notin \beta^i$, by performing operation $\ominus$ on the row R^i obtained by $\oplus$, we deduce that $\ominus(\beta^i) = \{c(i,j)\colon j = 1 \text{ or } 13 \leqslant j \leqslant 16\}$,

$$R^i = 0\,\lambda\,1\,1\,1\,1\,\lambda\,\lambda\,\lambda\,\lambda\,\lambda\,\lambda\,0\,0\,0\,0$$

Let $h_i = 7$. Since $\beta^i = \{c(i,j)\colon 2 \leqslant j \leqslant 12\}$ and $l(\alpha^i) = 3$, $r(\alpha^i) = 6$, by performing operation $\otimes$ on the row R^i obtained by $\oplus$ and $\ominus$, we get $\otimes(\alpha^i) = \{c(i,j)\colon 3 \leqslant j \leqslant 8\}$,

$$R^i = 0\,\lambda\,1\,1\,1\,1\,1\,1\,\lambda\,\lambda\,\lambda\,\lambda\,0\,0\,0\,0$$

We perform operation $\odot$ on the row R^i obtained by $\oplus$, $\ominus$ and $\otimes$ and assume that $h_i = 7$. Since $\alpha^i \neq \emptyset$ and $l(\alpha^i) = 3$, $r(\alpha^i) = 8$, we deduce $\odot(\beta^i) = \{c(i,j)\colon 2 \leqslant j \leqslant 9\}$,

$$R^i = 0\,\lambda\,1\,1\,1\,1\,1\,1\,\lambda\,0\,0\,0\,0\,0\,0\,0$$

If the filling operations are repeated, row R^i undergoes no further changes. However, if operations are performed on the columns, R^i can be changed and therefore further operations on that row may be necessary.

Example 3. It can be seen that if $\alpha^i = \emptyset$, that is, R^i does not contain any cells of α, then operations $\oplus$, $\ominus$ and $\odot$ do not change R^i, while operation $\otimes$ can put some cells into the kernel, as follows: if $\beta^i = \{c(i,j)\colon 2 \leqslant j \leqslant 15\}$, $\alpha^i = \emptyset$, and $h_i = 9$, then $\otimes(\alpha^i) = \{c(i,j)\colon 7 \leqslant j \leqslant 10\}$,

$$\otimes(0\,\lambda\,\lambda\,\lambda\,\lambda\,\lambda\,\lambda\,\lambda\,\lambda\,\lambda\,\lambda\,\lambda\,\lambda\,\lambda\,\lambda\,0) = 0\,\lambda\,\lambda\,\lambda\,\lambda\,\lambda\,1\,1\,1\,1\,\lambda\,\lambda\,\lambda\,\lambda\,\lambda\,0$$

Assuming that there exist a convex polyomino Λ that satisfies (H, V) so that $Q(i,j,k,h) \subseteq \Lambda$, kernel α and shell β are two sets such that

$$Q(i,j,k,h) \subseteq \alpha \subseteq \Lambda \subseteq \beta \subseteq R.$$

In this situation, if we perform operations $\oplus$, $\otimes$, $\ominus$ and $\odot$ on R^i and R_j, we can easily prove that:

$$\alpha^i \subseteq \oplus(\alpha^i) \subseteq \Lambda^i, \qquad \alpha_j \subseteq \oplus(\alpha_j) \subseteq \Lambda_j,$$
$$\alpha^i \subseteq \otimes(\alpha^i) \subseteq \Lambda^i, \qquad \alpha_j \subseteq \otimes(\alpha_j) \subseteq \Lambda_j,$$
$$\Lambda^i \subseteq \ominus(\beta^i) \subseteq \beta^i, \qquad \Lambda_j \subseteq \ominus(\beta_j) \subseteq \beta_j,$$
$$\Lambda^i \subseteq \odot(\beta^i) \subseteq \beta^i, \qquad \Lambda_j \subseteq \odot(\beta_j) \subseteq \beta_j.$$

Consequently, operations $\oplus$ and $\otimes$ expand α, while $\ominus$ and $\odot$ reduce β. It follows that the algorithm that reconstructs Λ initially sets $\alpha := Q(i, j, k, h)$, $\beta := R$ and performs the filling operations on R's rows and columns in the following order: $\oplus$, $\ominus$, $\otimes$, $\odot$ and sets $\alpha := \oplus(\alpha)$, $\beta := \ominus(\beta)$, $\alpha := \otimes(\alpha)$ and $\beta := \odot(\beta)$ at each row or column.

This directly gives the following corollary.

Corollary 3.1. *If the algorithm produces a kernel α and a shell β such that $\alpha \not\subset \beta$, then there is no convex polyomino Λ that satisfies (H, V) and such that $Q(i, j, k, h) \subseteq \Lambda$.*

Therefore, if we perform $\oplus$, $\ominus$, $\otimes$, $\odot$ and obtain $\alpha \not\subset \beta$ (i.e., the cells put into the kernel by $\oplus$ and $\otimes$ do not belong to the shell or the cells eliminated from the shell by $\ominus$ and $\odot$ belong to the kernel), there is no convex polyomino Λ that satisfies (H, V) and such that $Q(i, j, k, h) \subseteq \Lambda$. In this case, the attempts to reconstruct fail and the algorithm stops, chooses a different position of the feet (i_1, j_1, k_1, h_1) and repeats the whole procedure on $Q(i_1, j_1, k_1, h_1)$. As previously mentioned, the cells inserted in the kernel by $\oplus$ and $\otimes$ are labelled "1", while the cells eliminated from the shell by $\ominus$ and $\odot$ are labelled "0". As a result, the algorithm stops if a cell previously labelled "0" has to be labelled "1", or vice versa.

If we get α and β such that $\alpha = \beta$ and α is a convex polyomino, then $\alpha = \Lambda$. Otherwise, there is no a convex polyomino containing $Q(i, j, k, h)$ and that satisfies (H, V).

Let us now assume that neither of previous situations has occurred (i.e., there has been no failure and $\alpha \neq \beta$). The algorithm produces a kernel α and a shell β that cannot be further changed by filling operations and such that $\alpha \subset \beta$ (see Fig. 4). In this case, we are not yet able to say that there is a convex polyomino Λ that satisfies (H, V), because there are some example in which α and β are invariant with respect to operations $\oplus$, $\ominus$, $\otimes$, $\odot$, $\alpha \subset \beta$ and the polyomino Λ does not exist. Therefore, in order to establish the existence of Λ, another operation has to be performed. We define *evaluation* v of $\beta - \alpha$ as an arrangement of the "1" and "0" on cells of $\beta - \alpha$ such that all the cells of $\beta - \alpha$ are labelled; we denote $v(\alpha)$ the new kernel obtained. Determining the existence of Λ is thus linked to determining the existence of an evaluation v of $\beta - \alpha$ for which $v(\alpha)$ is a convex polyomino that satisfies (H, V). In order to describe and justify this further step, we must determine the shape of α and β when they are both invariant with respect to the filling operations and $\alpha \subseteq \beta$. We

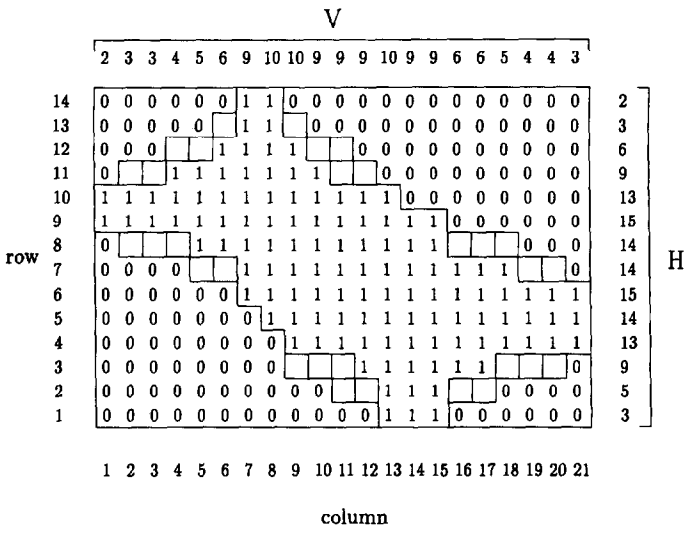

Fig. 4. Invariant α and β with respect to the filling operations and $\alpha \subset \beta$.

also have to prove that α and β are constructed in polynomial time. We begin by describing the shape of α and β.

Proposition 3.2. *Let $\alpha^i \neq \emptyset$. We perform the filling operations on R^i in the following order: $\oplus$, $\ominus$, $\otimes$, $\odot$. If we obtain α^i, β^i such that $\alpha^i \subset \beta^i$, then α^i, β^i are connected and invariant with respect to the filling operations. We also denote α^i and β^i with $\alpha^i = \{c(i,j)$: $l(\alpha^i) \leqslant j \leqslant r(\alpha^i)\}$, $\beta^i = \{c(i,j)$: $l(\beta^i) \leqslant j \leqslant r(\beta^i)\}$ and obtain*

$$r(\alpha^i) - l(\beta^i) + 1 = r(\beta^i) - l(\alpha^i) + 1 = h_i. \tag{3.1}$$

The same result holds if $\alpha^i = \emptyset$, $\otimes(\alpha^i) \neq \alpha^i$, $\alpha^i \subset \beta^i$ and the coherence operations are performed on R^i in the following order: $\otimes$, $\odot$.

If $\alpha_i = \emptyset$ and $\otimes(\alpha^i) = \alpha^i$, then the filling operations do not change α^i, β^i.

The same properties hold when operations $\oplus$, $\ominus$, $\otimes$, and $\odot$ are performed on column R_j.

Example 4. In Example 2, the cells of the kernel are $\alpha^i = \{c(i,3), c(i,6)\} \neq \emptyset$ and by performing the filling operations, we obtain

$$R^i = 0\,\lambda\,1\,1\,1\,1\,1\,1\,\lambda\,0\,0\,0\,0\,0\,0\,0$$

Therefore $\alpha^i = \{c(i,j)$: $3 \leqslant j \leqslant 8\}$, $\beta^i = \{c(i,j)$: $2 \leqslant j \leqslant 9\}$ and $r(\alpha^i) - l(\beta^i) + 1 = r(\beta^i) - l(\alpha^i) + 1 = h_i = 7$.

In Example 3, we have $\alpha^i = \emptyset$ and $\otimes(\alpha^i) \neq \emptyset$, and performing the coherence operations, we obtain

$$\otimes(0\,\lambda\,\lambda\,\lambda\,\lambda\,\lambda\,\lambda\,\lambda\,\lambda\,\lambda\,\lambda\,\lambda\,\lambda\,\lambda\,0) = 0\,\lambda\,\lambda\,\lambda\,\lambda\,\lambda\,1\,1\,1\,1\,\lambda\,\lambda\,\lambda\,\lambda\,\lambda\,0$$

Therefore, $\alpha^i = \{c(i,j)\colon 7 \leqslant j \leqslant 10\}$, $\beta^i = \{c(i,j)\colon 2 \leqslant j \leqslant 15\}$ and $r(\alpha^i) - l(\beta^i) + 1 = r(\beta^i) - l(\alpha^i) + 1 = h_i = 9$.

Remark 1. Let us take row R^i into consideration. From property (3.1) of the preceding proposition, we deduce that

$$l(\alpha^i) - l(\beta^i) = r(\beta^i) - r(\alpha^i),$$

that is, the number of $\beta^i - \alpha^i$ cells on the left of α^i coincides with the number of cells on the right of α^i. Moreover, since $\alpha^i \subseteq \beta^i$, we obtain that

$$r(\alpha^i) - l(\alpha^i) + 1 \leqslant h_i \leqslant r(\beta^i) - l(\beta^i) + 1.$$

In the first case of Example 4, we have $l(\alpha^i) - l(\beta^i) = r(\beta^i) - r(\alpha^i) = 1$ and $6 \leqslant h_i \leqslant 8$, and in the second $l(\alpha^i) - l(\beta^i) = r(\beta^i) - r(\alpha^i) = 5$ and $4 \leqslant h_i \leqslant 14$.

We now want to calculate the time required for constructing α and β invariant with respect to the filling operations. The algorithm performs the four operations on row R^i for $i = 1, 2, \ldots, m$ and then on column R_j for $j = 1, 2, \ldots, n$ and repeats this procedure until α and β cannot undergo any further changes. Performing $\oplus$, $\ominus$, $\otimes$ and $\odot$ on row R^i (column R_j) involves a computational cost of $O(m)$ ($O(n)$). Therefore, every time these operations are performed on all the rows and columns, $O(nm)$ operations are performed. The procedure is repeated less than nm times because it always inserts at least one cell in α or eliminates at least one cell from β. Since the number of R cells is equal to nm, the procedure cannot be repeated more than nm time. Consequently, α and β are constructed in $O(n^2m^2)$ time. In case of failure, that is, the algorithm generates α and β such that $\alpha \not\subset \beta$, the computational cost is still $O(n^2m^2)$.

At this point, we are able to describe the criterium that starts out from α and β, invariant with respect to the filling operations and such that $\alpha \subset \beta$, and which allows us to establish the existence of an evaluation v of $\beta - \alpha$ for which $v(\alpha)$ is a convex polyomino that satisfies (H, V). We first start by assuming that kernel α is a convex polyomino. We then examine the case in which α is a not convex polyomino and introduce some procedures that further expand the kernel and further reduce the shell. In this way, we produce a convex kernel and bring the problem back to the former case.

Let us then assume that kernel $\alpha \in (\mathbf{p}, \mathbf{v}, \mathbf{h})$. From Proposition 3.2, it follows that β is a convex polyomino and that:

Corollary 3.3. *Let α and β be invariant with respect to the filling operations and such that $\alpha \subset \beta$ and α is a convex polyomino. Assuming that $c(i,j) \in \beta^i - \alpha^i$, we have:*

if $l(\beta^i) \leqslant j \leqslant l(\alpha^i) - 1$, then $c(i, j + h_i) \in \beta^i - \alpha^i$;

if $r(\beta^i) \geqslant j \geqslant r(\alpha^i) + 1$, then $c(i, j - h_i) \in \beta^i - \alpha^i$.

Analogously, assuming that $c(i,j)\in\beta_j-\alpha_j$, *we have*:
if $d(\beta_j)\leqslant i\leqslant d(\alpha_j)-1$ *then* $c(i+v_j,j)\in\beta_j-\alpha_j$;
if $u(\beta_j)\geqslant i\geqslant u(\alpha_j)+1$ *then* $c(i-v_j,j)\in\beta_j-\alpha_j$.

Proof. Let $c(i,j)\in\beta^i-\alpha^i$ with $l(\beta^i)\leqslant j\leqslant l(\alpha^i)-1$. Since α is a convex polyomino, we have $\alpha^i\neq\emptyset$. By performing the filling operations on α^i, α^i does not change and so from Proposition 3.2, we have

$$r(\alpha^i)-l(\beta^i)+1=r(\beta^i)-l(\alpha^i)+1=h_i.$$

However,

$$l(\beta^i)+h_i\leqslant j+h_i\leqslant l(\alpha^i)-1+h_i$$

and thus we get

$$r(\alpha^i)+1\leqslant j+h_i\leqslant r(\beta^i),$$

from the preceding condition. We use the same procedure for the other case. □

Let $c(i,j)\in\beta^i-\alpha^i$ with $l(\beta^i)\leqslant j<l(\alpha^i)$. From Corollary 3.3, it follows that $c(i,j+h_i)\in\beta^i-\alpha^i$. If cell $c(i,j)$ is labelled "1" ("0"), then by performing operation $\odot$ ($\otimes$) on R^i, we label cell $c(i,j+h_i)$ "0" ("1").

Example 5. If R^i coincides with the row in Example 3 and cell $c(i,3)$ is labelled "1", since $h_i=9$, we label cell $c(i,12)$ "0" by performing $\odot$ on R^i:

$$\odot(0\lambda\underline{1}\lambda\lambda\lambda1111\lambda\lambda\lambda\lambda\lambda0)=0\lambda1\lambda\lambda\lambda1111\lambda\underline{0}0000$$

Vice versa, if cell $c(i,3)$ is labelled "0", we label cell $c(i,12)$ "1" by performing $\odot$ on R^i:

$$\otimes(0\lambda\underline{0}\lambda\lambda\lambda1111\lambda\lambda\lambda\lambda\lambda0)=0\lambda0\lambda\lambda\lambda1111\underline{1}\lambda\lambda\lambda0$$

Analogously, if $c(i,j)\in\beta^i-\alpha^i$ with $r(\alpha^i)<j\leqslant r(\beta^i)$, then $c(i,j-h_i)\in\beta^i-\alpha^i$ and if cell $c(i,j)$ is labelled "1" ("0"), then we label cell $c(i,j-h_i)$ "0" ("1") by performing operation $\odot$ ($\otimes$) on R^i. The same holds for column R_j. Therefore, there is a relationship between the cells belonging to $\beta-\alpha$ caused by the coherence operations. If $c(i,j)\in\beta-\alpha$, then there are two cells $c(i,j')$, $c(i',j)\in\beta-\alpha$ such that $|j-j'|=h_i$, $|i-i'|=v_j$. If $c(i,j)$ is labelled "1", then $c(i,j')$, $c(i',j)$ are labelled "0", while if $c(i,j)$ is labelled "0", then $c(i,j')$, $c(i',j)$ are labelled "1". This similarity allows us to introduce the following definition.

Definition 3.4. We call *graph on* $\beta-\alpha$ the graph $G=(V,E)$ such that:
$V=\beta-\alpha$;
$(c(i,j),c(i',j'))\in E$ *iff* $(i=i'$ *and* $|j-j'|=h_i)$ *or* $(j=j'$ *and* $|i-i'|=v_j)$.

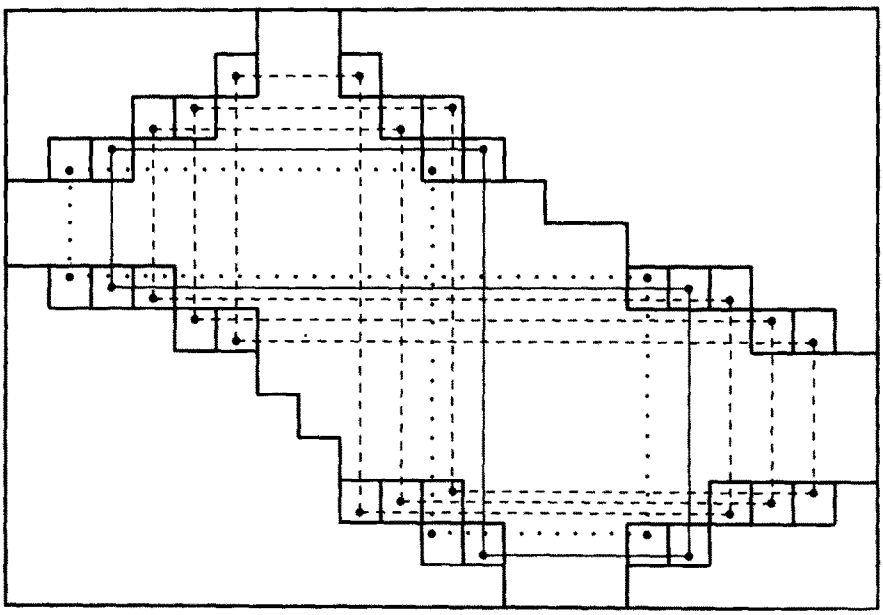

Fig. 5. Graph on $\beta - \alpha$ made up of 3 unjoined cycles.

From definition of G, we have:

the degree of each of G's vertices is equal to 2;

if $(a, b) \in E$, then $a \neq b$;

every component of G contains an even number of vertices.

Fig. 5 is an example of the graph on $\beta - \alpha$.

It follows that G is a cycle or a union of disjoint cycles, each of which contains at least 4 vertices. The graph in Fig. 5 is made up by 3 disjoint cycles. Determining a cycle in G requires a linear time with respect to the number of its vertices. Since the cycles are disjoint, the time required for constructing G is $O(nm)$. Every vertex of the graph is a cell of $\beta - \alpha$ that can be labelled "1" or "0" and so the vertex can assume two different states. We can say that $c(i, j) \in V$ has value 1 or 0 depending on how $c(i, j)$ is labelled. It follows from definition of G that, if a vertex has value $x \in \{0, 1\}$, then the values of the vertices belonging to the cycle containing $c(i, j)$ are univocally determined by x. In fact, if $(c(i, j), c(i', j)), (c(i, j), c(i, j')) \in E$, then the value of $c(i', j)$ and $c(i, j')$ is $\bar{x}$. Consequently, a cycle in G can only assume two states and, in each state half of its vertices have value 1 and the other half 0, and the vertices of each cycle edge have value 1 and 0 (see Fig. 6). We indicate G's cycles by $\{(V_1, E_1), (V_2, E_2), \ldots, (V_k, E_k)\}$. Assuming that $c(i, j) \in V_h$, we label the cell with the boolean variable x_h and, arting out from $c(i, j)$, we go through the cycle labelling the vertices x_h or $\bar{x}_h$, according to whether the number of edges between the vertices and $c(i, j)$ is even or odd. The labelling can be done during the graph's construction (see Fig. 6). By assigning some

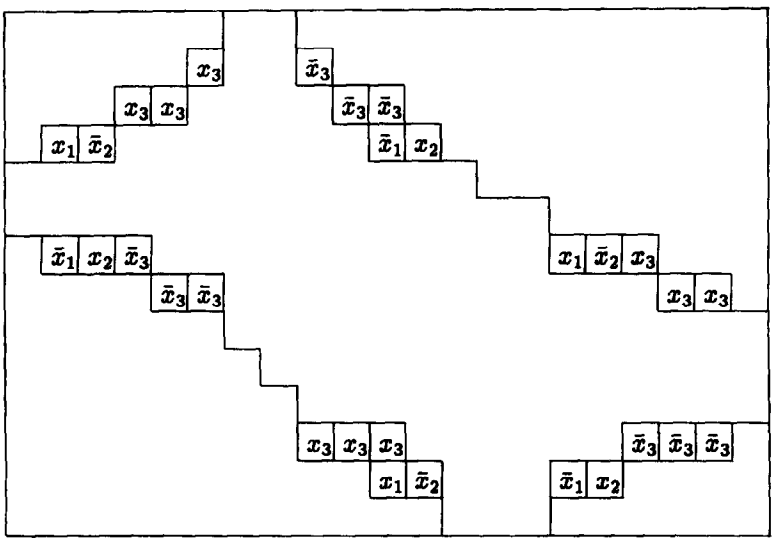

Fig. 6. Possible states of G's cycles.

values to the boolean variables $x_1, x_2, \ldots, x_k$, we obtain an evaluation of the cycles in G because each assignment determines the state of a cycle. An evaluation of G's cycles coincides with evaluation v of $\beta - \alpha$ and it therefore originates a kernel $v(\alpha)$.

Proposition 3.5. *For each evaluation v of the cycles in G, $v(\alpha)$ satisfies (H, V).*

Proof. Let v be an evaluation of the cycles in G. In order to prove that $v(\alpha)$ satisfies (H, V), we have to show that $|v(\alpha)^i| = h_i$ and $|v(\alpha)_j| = v_j$. From Remark 1 it follows that

$$l(\alpha^i) - l(\beta^i) = r(\beta^i) - r(\alpha^i) = \frac{|\beta^i - \alpha^i|}{2}.$$

By evaluating the cycles in G, if $c(i,j) \in \beta^i - \alpha^i$ and has value x, then the cell $c(i,j') \in \beta^i - \alpha^i$, such that $|j - j'| = h_i$, has value $\bar{x}$. Therefore, the number of cells in $\beta^i - \alpha^i$ having value x coincides with those having value $\bar{x}$ and, consequently, it coincides with $l(\alpha^i) - l(\beta^i)$. It follows that the number of cells in $v(\alpha)^i$ labelled "1" is:

$$|v(\alpha)^i| = l(\alpha^i) - l(\beta^i) + |\alpha^i| = r(\alpha^i) - l(\beta^i) + 1.$$

However, it follows from Proposition 3.2 that $r(\alpha^i) - l(\beta^i) + 1 = h_i$ and therefore $|v(\alpha)^i| = h_i$. By the same way, we show that $|v(\alpha)_i| = v_j$. □

From Proposition 3.5, it follows that:

Corollary 3.6. *There is a convex polyomino Λ that satisfies (H, V) and such that $Q(i, j, k, h) \subset \Lambda$ iff there is an evaluation v of G's cycles for which $v(\alpha)$ is a convex polyomino.*

Consequently, determining the existence of Λ is linked to determining an evaluation v such that $\forall i \in [1..m]$ and $\forall j \in [1..n]$ $v(\alpha)^i$ and $v(\alpha)_j$ are connected. Assuming that $\alpha \subset \beta$ and the kernel α is a convex polyomino, we obtain that the sets

$W(\beta - \alpha) = \{c(i,j) \in \beta - \alpha : i \in [1..m],\ l(\beta^i) \leqslant j \leqslant l(\alpha^i)\}$,

$S(\beta - \alpha) = \{c(i,j) \in \beta - \alpha : j \in [1..n],\ d(\beta_j) \leqslant i \leqslant d(\alpha_j)\}$

are made up of the cells in $\beta - \alpha$ that are to the west of P_N and P_S and to the south of P_W and P_E, respectively. Let R^i be any row containing $W(\beta - \alpha)$'s cells; that is, $W(\beta^i - \alpha^i) \neq \emptyset$. In this case the number of cells in $W(\beta^i - \alpha^i)$ is $p = l(\alpha^i) - l(\beta^i)$. Let us cross over R^i from left to right and denote the boolean variables associated to the $W(\beta^i - \alpha^i)$'s cells by $Z = (z_1, z_2, \ldots, z_p)$ (that is, $z_i \in \{x_1, \bar{x}_1, \ldots, x_k, \bar{x}_k\}$). An evaluation v on G's cycles determines an evaluation of Z and therefore of $W(\beta^i - \alpha^i)$. The evaluation v originates a connected row $v(\alpha^i)$ iff there is an integer $q \in [1..(p+1)]$ such that

$$z_h = \begin{cases} 0 & \text{if } h < q, \\ 1 & \text{if } h \geqslant q. \end{cases} \tag{3.2}$$

If this condition holds, then v labels the cells in $W(\beta^i - \alpha^i)$ as

$$R^i = 0 \ldots 0\ z_1 \ldots z_{q-1} z_q \ldots z_p\ 1 \ldots 1\ \bar{z}_1 \ldots \bar{z}_{q-1} \bar{z}_q \ldots \bar{z}_p\ 0 \ldots 0$$

$$v(R^i) = 0 \ldots 0\ 0 \ldots 0\ 1 \ldots 1\ 1 \ldots 1\ 1 \ldots 1\ 0 \ldots 0\ 0 \ldots 0$$

and therefore $v(\alpha)^i$ is connected.

Vice versa, if $v(\alpha)^i$ is connected, then condition (3.2) obviously holds. We therefore say that an evaluation v of G's cycles is *connected* on R^i if the values assumed by Z's elements satisfy condition (3.2).

Lemma 3.7. *An evaluation v of G's cycles is connected on R^i iff the values assumed by Z's elements satisfy the following collection F of clauses over X:*

$$F \stackrel{\text{def}}{=} (\bar{z}_1 \vee z_2) \wedge (\bar{z}_2 \vee z_3) \wedge \cdots \wedge (\bar{z}_{p-1} \vee z_p). \tag{3.3}$$

Proof. Let v be a connected evaluation on R^i. Since, condition (3.2) holds, given clause $\bar{z}_{h-1} \vee z_h$, there are three possibilities:

if $h < q$ then $z_{h-1} = 0$ and $z_h = 0$;

if $h = q$ then $z_{h-1} = 0$ and $z_h = 1$;

if $h > q$ then $z_{h-1} = 1$ and $z_h = 1$.

In each case, clause $\bar{z}_{h-1} \vee z_h$ is equal to 1 and therefore F is equal to 1.

Vice versa, if the values of Z's elements satisfy F (that is, F is equal to 1 for these values), then clause $\bar{z}_{h-1} \vee z_h$ is equal to 1 for each $h \in [2..p]$. If we make the absurd assumption that the evaluation v of G's cycles is not connected on R^i, then there is at

least one $q \in [2..p]$ such that $z_{q-1} = 1$ and $z_q = 0$. But this means that $\bar{z}_{q-1} \vee z_q = 0$ and that is absurd. Therefore evaluation v is connected on R^i. □

We use the same procedure for $S(\beta - \alpha)$. Let us cross over R_j from the bottom towards the top and denote the boolean variables associated to the $S(\beta^i - \alpha^i)$ cells by $Z = (z_1, z_2, \ldots, z_p)$. We have that $v(\alpha)_j$ is connected iff the values assumed by Z's elements satisfy the clause collection F (3.3) over X.

We denote $z(i,j)$ the variable associated to cell $c(i,j) \in W(\beta - \alpha) \cup S(\beta - \alpha)$ and denote F^i and F_j the following clause collections over X:

$F^i \stackrel{\text{def}}{=} (\bar{z}(i, l(\beta^i)) \vee z(i, l(\beta^i) + 1) \wedge (\bar{z}(i, l(\beta^i) + 1) \vee z(i, l(\beta^i) + 2) \wedge \ldots \wedge (\bar{z}(i, l(\alpha^i) - 2) \vee z(i, l(\alpha^i) - 1))$;

$F_j \stackrel{\text{def}}{=} (\bar{z}(d(\beta^i), j) \vee z(d(\beta^i) + 1), j) \wedge (\bar{z}(d(\beta^i) + 1, j) \vee z(d(\beta_j) + 2), j) \wedge \ldots \wedge (\bar{z}(d(\alpha_j) - 2, j) \vee z(d(\alpha_j) - 1, j))$;

Therefore, $z(i,j) \in X = \{x_1, x_2, \ldots, x_k\}$ and F^i and F_j are the clause collections over X associated to row $W(\beta^i - \alpha^i)$ and column $S(\beta_j - \alpha_j)$, respectively. If $W(\beta^i - \alpha^i) = \emptyset$ ($S(\beta_j - \alpha_j) = \emptyset$), then we set $F^i = 1$ ($F_j = 1$). From Corollary 3.6 and Lemma 3.7, it follows that:

Proposition 3.8. *There is a convex polyomino Λ that satisfies (H, V) and such that $Q(i,j,k,h) \subset \Lambda$ iff there is an evaluation of X's variables that satisfies the following clause collection over X:*

$$C \stackrel{\text{def}}{=} \left(\bigwedge_{i=1}^{m} F^i \right) \wedge \left(\bigwedge_{j=1}^{n} F_j \right).$$

Therefore, if the kernel and the shell obtained by the filling operations are such that $\alpha \subset \beta$ and α is a convex polyomino, then the problem of determining the existence of a convex polyomino that satisfies (H, V) is linked to the 2-SATISFIABILITY problem (also referred to as 2SAT [14]). In fact, C is a clause collection over $X = \{x_1, x_2, \ldots, x_k\}$ made up of clauses containing two variables. The 2SAT problem can be solved in linear time with respect to the number of C's clauses and X's variables (see [1]). The number of X's variables coincides with the number of G's cycles and is therefore less than $nm/4$. The number of F^i $\{F_j\}$'s clauses is less than $n/2$ $(m/2)$ and so the number of C's clauses is less than nm. As a result, the time needed to find out if there is an assignment of values of X's variables that satisfies C is less than $O(nm)$. Moreover, the number of operations needed for constructing C is less than nm. Therefore, if we start out with α and β invariant with respect to the filling operations and such that $\alpha \subset \beta$ and α is a convex polyomino, then we can find out if (H, V) is satisfiable in $(\mathbf{p}, \mathbf{v}, \mathbf{h})$ in $O(nm)$ time.

Let us now assume that the kernel α generated by the filling operations is not a convex polyomino. Our method starts out with α and β satisfying the condition

(1) α and β are invariant with respect to the filling operations, $\alpha \subset \beta$ and $\alpha \notin (\mathbf{p}, \mathbf{v}, \mathbf{h})$,

and performs a further expansion of α and reduction of β such that the problem falls into one of the previous cases. In other words, α and β are generated such that:

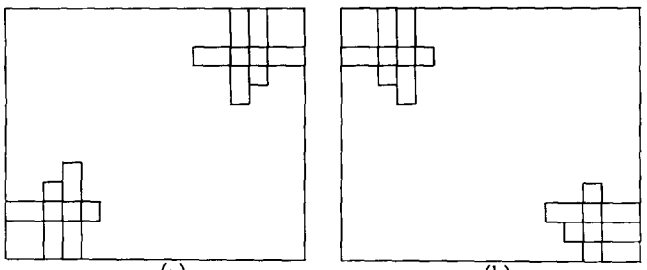

Fig. 7. Positions of the feet that allow us to obtain a kernel $\alpha \notin (\mathbf{p}, \mathbf{v}, \mathbf{h})$.

(2) $\alpha \not\subset \beta$, or

(3) $\alpha = \beta$, or

(4) α and β are invariant with respect to the filling operations, $\alpha \subset \beta$ and $\alpha \in (\mathbf{p}, \mathbf{v}, \mathbf{h})$.

If there is a convex polyomino Λ that satisfies (H, V) and such that $Q(i, j, k, h) = (P_S(i), P_E(j), P_N(k), P_W(h)) \subset \Lambda$, then it follows from the definition of convex polyomino that two pairs of consecutive feet have a nonempty intersection. In other words,

$$(P_S(i) \cap P_E(j) \neq \emptyset \quad \text{and} \quad P_N(k) \cap P_W(h) \neq \emptyset)$$

or

$$(P_E(j) \cap P_N(k) \neq \emptyset \quad \text{and} \quad P_W(h) \cap P_S(i) \neq \emptyset) \tag{3.4}$$

Therefore, if this condition does not hold, the above mentioned polyomino Λ does not exist. Consequently, once the position $Q(i, j, k, h)$ of feet has been chosen, the algorithm checks to see if condition (3.4) holds and, if so, performs the filling operations. Given our definition of the operations, the only positions of the feet able to originate α and β satisfying condition (1) are the ones shown in Fig. 7. Moreover, from Proposition 3.2, it follows that, if $\alpha^i \neq \emptyset$, then α^i and β^i are connected (the same holds for the columns). Therefore, if we start out with the positions illustrated in Fig. 7 and perform the filling operations we obtain α and β satisfying condition (1), then α is made up of two or more disjoint convex polyominoes. Initially, $\alpha := Q(i, j, k, h)$ and, therefore, these two polyominoes, denoted Λ_1 and Λ_2, contain the feet. If we start out with the positions shown in Fig. 7(a), then $P_S(i)$, $P_W(h)$ belong to Λ_1 and $P_E(j)$, $P_N(k)$ belong to Λ_2. Let us assume Λ_1 and Λ_2 be the following:

$\Lambda_1 = \{c(i, j) \in \alpha\colon 1 \leqslant i \leqslant p_1,\ 1 \leqslant j \leqslant q_1\}$;

$\Lambda_2 = \{c(i, j) \in \alpha\colon p_2 \leqslant i \leqslant m,\ q_2 \leqslant j \leqslant n\}$.

Let r_1 and s_1 (r_2 and s_2) be the positions of Λ_1 (Λ_2)'s row and column as shown in Fig. 8. If there is a polyomino Λ that satisfies (H, V) and such that $Q(i, j, k, h) \subseteq \Lambda$, then the polyominoes Λ_1 and Λ_2 are part of Λ and it follows from Λ's convexity that sets

$\bar{\Lambda}_1 = \{c(i, j)\colon r_1 < i \leqslant p_1,\ s_1 < j \leqslant q_1\}$;

$\bar{\Lambda}_2 = \{c(i, j)\colon p_2 \leqslant i < r_2,\ q_2 \leqslant j < s_2\}$,

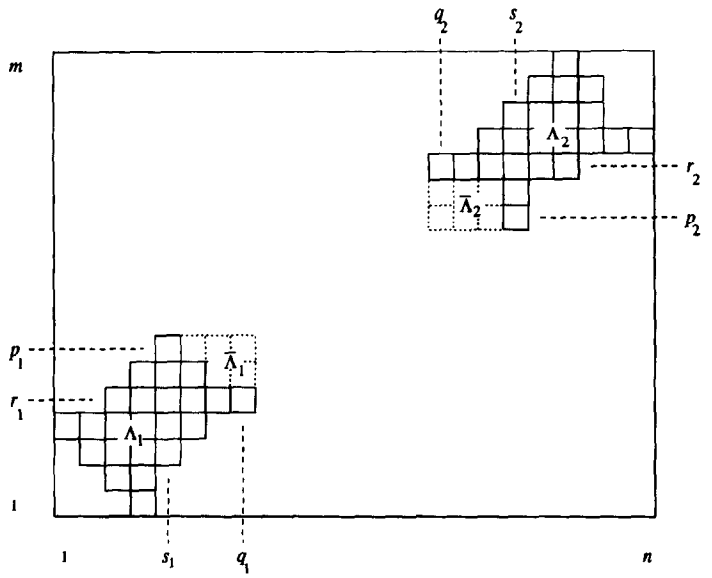

Fig. 8. Convex polyominoes Λ_1 and Λ_2 that contain $P_S(i)$, $P_W(h)$ and $P_E(j)$, $P_N(k)$, respectively.

are contained in Λ. The algorithm thus expands the kernel by inserting these sets in α. If there are no failures in this expansion (that is, we obtain $\alpha \subseteq \beta$), we repeat the filling operations and the new generated α's and β's satisfy one of the conditions (1)–(4). If we obtain that α and β still satisfy condition (1), then the kernel is expanded in the same way as above, and so on. It is worth noting, however, that the filling operations may produce α and β satisfying condition (1) with $\bar{\Lambda}_1$ and $\bar{\Lambda}_2$ already belonging to α. In this case, if $c(p_1+1, q_1)$ and $c(p_1, q_1+1)$ do not belong to $\beta-\alpha$, there is no convex polyomino Λ that satisfies (H, V) and such that $Q(i,j,k,h) \subseteq \Lambda$. If only one of the two cells belongs to $\beta-\alpha$, then that cell is inserted in α and the filling operations are repeated. If both cells belong to $\beta-\alpha$, further expansion of the kernel is more complex. Since Λ_1 is a convex polyomino, then Corollary 3.3 holds for $\beta^i-\alpha^i$ and $\beta_j-\alpha_j$ with $i\in[1..p_1]$ and $j\in[1..q_1]$. Therefore, if:

$A \stackrel{\text{def}}{=} \{c(i,j)\in\beta-\alpha\colon 1\leqslant i\leqslant p_1,\ 1\leqslant j\leqslant q_1\}$,
$B_1 \stackrel{\text{def}}{=} \{c(i,j)\in\beta-\alpha\colon p_1<i<m,\ 1\leqslant j\leqslant q_1\}$,
$B_2 \stackrel{\text{def}}{=} \{c(i,j)\in\beta-\alpha\colon 1\leqslant i\leqslant p_1,\ q_1<j<n\}$.

then:

if $c(i,j)\in A$, there are $c(i,j'), c(i',j)\in\beta-\alpha$ such that $|j-j'|=h_i$, $|i-i'|=v_j$,
if $c(i,j)\in B_1$, then there is $c(i',j)\in\beta-\alpha$ such that $|i-i'|=v_j$;
if $c(i,j)\in B_2$, then there is $c(i,j')\in\beta-\alpha$ such that $|j-j'|=h_i$.

We can thus define a graph $G=(V,E)$ on $\beta^i-\alpha^i$ and $\beta_j-\alpha_j$ for $i\in[1..p_1]$ and $j\in[1..q_1]$ (that is, with $V=A\cup B_1\cup B_2$), as previously done when α was a convex polyomino (see Fig. 9). It can be seen that G's vertices are of 1 or 2 degree. In fact, if

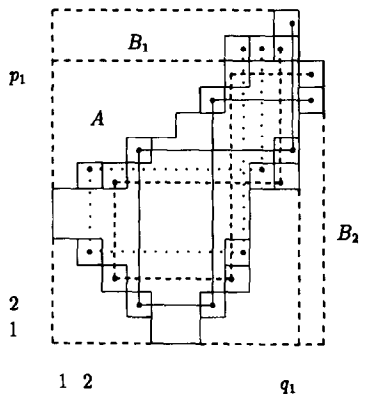

Fig. 9. Graph on $A \cup B_1 \cup B_2$ made up of three disjoint open walks.

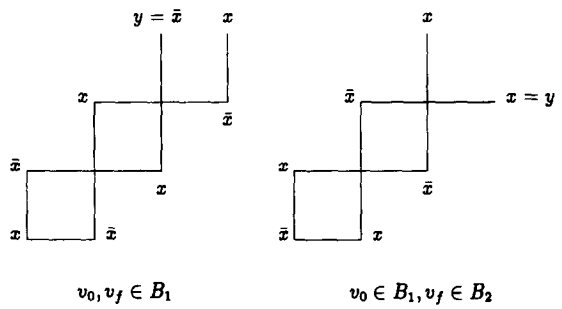

Fig. 10. Walk C on graph G.

$c(i,j) \in A$, then $c(i,j)$'s degree is 2, while if $c(i,j) \in B_1 \cup B_2$, then $c(i,j)$'s degree is 1. Consequently, it follows from Definition 3.4 that G is a union of open disjoint walks all having their ends in $B_1 \cup B_2$. These walks are made up of alternating sequences of horizontal and vertical steps (that is, each horizontal step is followed by a vertical step and vice versa). Let us assume that $v_0 \in B_i$ and $v_f \in B_j$ are the first and last vertices of a walk C in G, respectively. If v_0 and v_f belong both to B_1 (or B_2) then C's first and last steps are both vertical or both horizontal and therefore C is made up of an odd number of steps. On the contrary, C's first and last steps are of different kind and so C is made up of an even number of steps. As a result, if we assume that x and y are the boolean variables associated to v_0 and v_f and consider the fact that each vertex v_k in the walk is associated with x or $\bar{x}$ according to whether the number of steps between v_k and v_0 is even or odd, we have $y = \bar{x}$ in the first else $y = x$ in the latter (see Fig. 10).

Let us now take cell $c(p_1 + 1, q_1) \in B_1$ into consideration. This cell is the starting vertex of a walk in the graph whose final vertex v_f belongs to B_1 or B_2. If $v_f \in B_1$ then $y = \bar{x}$, and so if $x = 0$, then $y = 1$. But the polyomino A that satisfies (H, V) has to be

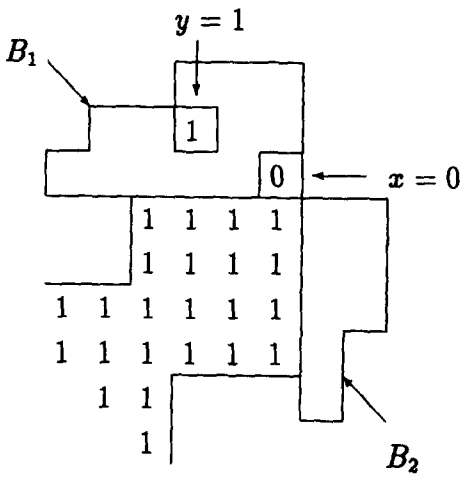

Fig. 11. G's walk having $v_f \in B_1$ that violates Λ's convexity.

convex and so this case is not acceptable (see Fig. 11). Consequently, if $v_f \in B_1$ and Λ does exist, then $x = 1$ and so cell $c(p_1 + 1, q_1)$ is labelled "1".

Analogously, if u_f is the last vertex of a graph's walk starting from $c(p_1, q_1 + 1)$ and $u_f \in B_2$, then cell $c(p_1, q_1 + 1)$ is labelled "1".

Let us now assume that $v_f \in B_2$ and $u_f \in B_1$. We denote x_1 and x_2 the variables associated to $c(p_1 + 1, q_1)$ and $c(p_1, q_1 + 1)$ and y_1 and y_2 the variables associated to v_f and u_f, respectively. Therefore, $y_1 = x_1$ and $y_2 = x_2$. In this case variables x_1 and x_2 are such that $x_1 = 1$ and $x_2 = 1$. In fact, if $x_1 = 0$ and a polyomino Λ that satisfies (H, V) exists, then $x_2 = 1$. Therefore, $y_2 = 1$ and $x_1 = 0$ (see Fig. 12). But the polyomino Λ that satisfies (H, V) has to be convex and so this case is not acceptable. Cells $c(p_1, q_1 + 1)$ and $c(p_1 + 1, q_1)$ are thus labelled "1".

It follows that:

if $v_f \in B_1$, then cell $c(p_1 + 1, q_1)$ is labelled "1";

if $u_f \in B_1$, then cell $c(p_1, q_1 + 1)$ is labelled "1";

if $v_f \in B_2$ and $u_f \in B_1$, then cell $c(p_1, q_1 + 1)$ and $c(p_1 + 1, q_1)$ are labelled "1".

The same holds when the feet's initial position is the one shown in Fig. 7(b).

In other words, if the filling operations generate α and β satisfying condition (1), then the procedure EXP used for expanding the kernel is made up of two possible alternatives:

(a) If $\bar{\Lambda}_1$ and $\bar{\Lambda}_2$ belong to α, the walks in the graph starting from cells $c(p_1, q_1 + 1)$ and $c(p_1 + 1, q_1)$ are constructed and these two cells are inserted in α according to the above mentioned procedure,

(b) If $\bar{\Lambda}_1$ and $\bar{\Lambda}_2$ do not belong to α, these areas are inserted in α.

After this, the filling operations are repeated and α and β satisfying one of the conditions (1)–(4) are obtained with a finite number steps.

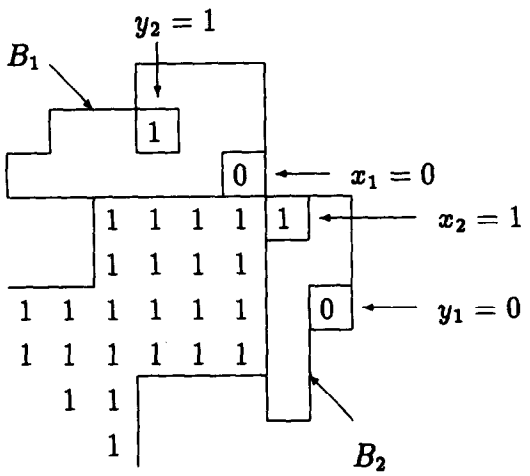

Fig. 12. G's walks having $v_f \in B_2$ and $u_f \in B_1$ violating Λ's convexity.

Let us now establish the number of operations necessary for obtaining α and β satisfying one of the conditions (2)–(4). Let w_1 and l_1 (w_2 and l_2) be, respectively, the width and length of Λ_1 (Λ_2). We perform the EXP procedure and if we obtain $w_1 + w_2 > n$ or $l_1 + l_2 > m$, then, by means of the filling operations we generate α and β satisfying one of the conditions (2)–(4). If we perform the EXP alternative (a), then cell $c(p_1 + 1, q_1)$ or $c(p_1, q_1 + 1)$ is added to polyomino Λ_1 and therefore w_1 or l_1 increases by 1. If we perform both the EXP alternative (b) and the filling operations and w_1, l_1 do not change, then if we repeat EXP, we perform alternative (a). It follows that the maximum number of times EXP and the filling operations are performed to obtain α and β satisfying one of conditions (2)–(4) is less than $2(n + m)$.

The walks built by alternative (a) are disjoint and their construction requires a linear time with respect to the number of vertices, while the number of operations required for inserting $\bar{\Lambda}_1$ and $\bar{\Lambda}_2$ in α is linear with respect to the number of cells inserted. It follows that EXP's complexity is less than, or equal to, $O(nm)$ and therefore the number of operations performed by EXP to obtain α and β satisfying one of conditions (2)–(4) is less than, or equal to, $O(n^2m + nm^2)$.

At the end of each EXP procedure, the filling operations are performed until we obtain α and β invariant. This invariance is obtained when the operations are performed on each row and column and α and β do not change. Consequently, the number of times that the filling operations are performed on all rows and columns is less than $2(n + m) + nm$. Since we perform $O(nm)$ operations when performing $\oplus$, $\ominus$, $\otimes$ and $\odot$ on all rows and columns, by starting out with $Q(i, j, k, h)$ satisfying condition (3.4), we define α and β that satisfy one of conditions (2)–(4) in $O(n^2m^2)$ time.

However, as previously seen, if α and β satisfy one of conditions (2)–(4), $O(nm)$ time is required to find out if (H, V) is satisfiable in $(\mathbf{p}, \mathbf{v}, \mathbf{h})$. Consequently, since it can be determined in linear time that $Q(i, j, k, h)$ satisfies condition (3.4), the existence of

a convex polyomino A that satisfies (H, V) such that $Q(i, j, k, h) \subseteq A$ is verified in $O(n^2m^2)$ time.

Since the number of possible positions of the feet $Q(i, j, k, h)$ is less than n^2m^2, we deduce that:

Theorem 3.9. *The RSP problem on* **(p, v, h)** *can be solved in a time that is less than or equal to* $O(n^4m^4)$.

4. RSP problem on (p, v), (p, h), (v) and (h)

Let us examine the following problem:

Numerical matching with target sums (NMTS)

Instance: Disjoint sets X and Y, each containing n elements, a function $s: X \cup Y \to \mathbb{N}$ polynomial in n, and a vector $V = (v_1, v_2, \ldots, v_n) \in \mathbb{N}^n$.

Question: Can set $X \cup Y$ be partitioned into n disjoint sets $A_1, A_2, \ldots, A_n$ such that $\forall i \in [1..n]$ $A_i = \{x, y\}$ with $x \in X$ and $y \in Y$, and $s(x) + s(y) = v_i$?

The NMTS problem is NP-complete, because the *Numerical 3-dimensional matching* is NP-complete and there is polynomial transformation that reduces it to NMTS problem (see [14]).

Theorem 4.1. *The RSP problem on class* **(p, v)** *is* NP*-complete.*

Proof. It is easy to see that RSP on **(p, v)** belongs to NP, since a nondeterministic algorithm needs only guess a column-convex polyomino A and check in polynomial time that A satisfies (H, V).

We give a polynomial time reduction from NMTS to RSP. Let an arbitrary instance of NMTS be given by:

two disjoint sets with n elements: $X = \{x_1, x_2, \ldots, x_n\}$, $Y = \{y_1, y_2, \ldots, y_n\}$,

a function $s: X \cup Y \to \mathbb{N}$ polynomial in n,

a vector $V = (v_1, v_2, \ldots, v_n) \in \mathbb{N}^n$.

We must construct two vectors $H' \in \mathbb{N}^m$ and $V' \in \mathbb{N}^n$ in such a way that (H', V') is satisfiable in **(p, v)** iff the set $X \cup Y$ can be partitioned into n disjoint sets $A_1, A_2, \ldots, A_n$ such that $\forall i \in [1..n]$ $A_i = \{x, y\}$ with $x \in X$ and $y \in Y$, and $s(x) + s(y) = v_i$.

Assuming that $u_i = s(x_i)$ and $d_i = s(y_i)$, for $1 \leqslant i \leqslant n$, two vectors of positive integers $U = (u_1, u_2, \ldots, u_n)$ and $D = (d_1, d_2, \ldots, d_n)$ are obtained from X and Y. Each element u_i (d_i) is associated with a column containing u_i (d_i) cells. We join these columns as shown in Fig. 13(a). We define the pair of vectors (H', V') (that is, the RSP instance) as follows:

$H' \stackrel{\text{def}}{=} (h'_1, h'_2, \ldots, h'_m)$, where h'_i is the number of cells in the ith row of the polyomino in Fig. 13,

$V' \stackrel{\text{def}}{=} V$.

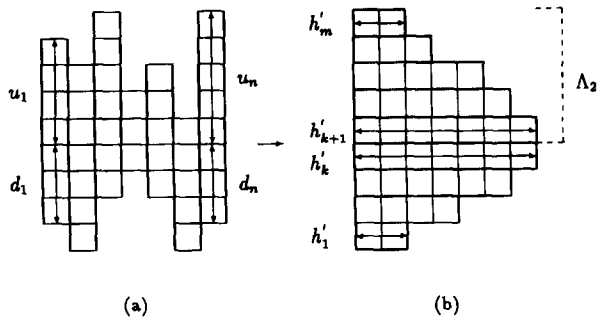

Fig. 13. Column-convex polyomino obtained from the columns associated to U and D.

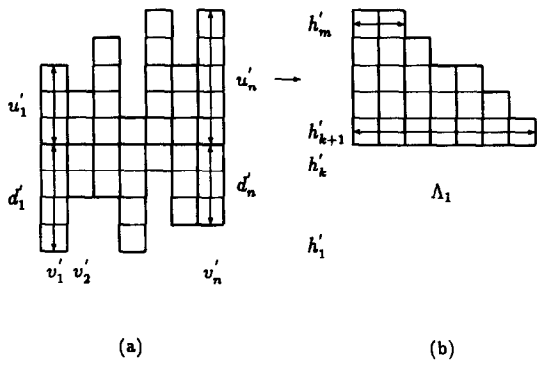

Fig. 14. Column-convex polyomino that satisfies (H', V').

Since, the number m of polyomino's rows is such that

$$m = \max U + \max D = \max\{s(x): x \in X\} + \max\{s(y): y \in Y\},$$

the function s is polynomial in n and the number of polyomino's columns is equal to n, then pair (H', V') can be constructed in polynomial time in the size of given NMTS's instance. It is worth noting that there is a $k \in [1..m]$ such that $h'_k = h'_{k+1} = n$, because all of U and D's elements are positive integers (see Fig. 13(b)).

Let us assume that (H', V') is satisfiable on $(\mathbf{p}, \mathbf{v})$. There is a column-convex polyomino Λ such that the ith row projection and the jth column projection are equal to h'_i and v'_j, respectively. Since $h'_k = h'_{k+1} = n$, Λ's kth and $(k+1)$th rows are n long. Moreover, Λ is vertically convex and so we can associate a vector $D' = (d'_1, d'_2, \ldots, d'_n) \in \mathbb{N}^n$ to Λ's columns made up of its first k rows and a vector $U' = (u'_1, u'_2, \ldots, u'_n) \in \mathbb{N}^n$ to Λ's columns made up of the subsequent $m - k$ rows (see Fig. 14). Polyomino Λ satisfies (H', V') and so its ith column is v'_i long and therefore vectors U' and D' are such that $\forall i \in [1..n]\ u'_i + d'_i = v_i$. We now prove that U' and D' are a permutation of U and D, respectively. Let us examine polyomino Λ_1 obtained from the columns associated to U' lined up according to their decreasing length (see Fig. 14(b)). Since Λ satisfies (H', V'), the length of Λ_1's jth row is equal to h'_{k+j}. Let us

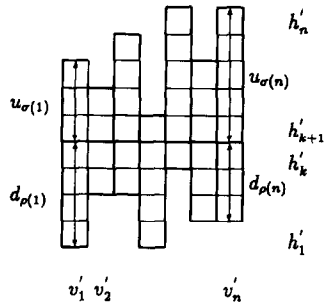

Fig. 15. Column-convex polyomino obtained by joining the columns related to $\sigma(U)$ and $\rho(D)$.

examine polyomino Λ_2 obtained by the columns associated to U lined up according to their decreasing length (see Fig. 13(b)). According to our definition of H', the length of Λ_2's jth row coincides with h'_{k+j} and so $\Lambda_1 = \Lambda_2$. As a result, U' is a permutation of U. We show that D' is a permutation of D in the same way. Therefore, there is a permutation σ of U and a permutation ρ of D such that $u_{\sigma(i)} + d_{\rho(i)} = v_i$ for $i = 1, 2, \ldots, n$. Consequently, set $X \cup Y$ can be partitioned into n disjoint sets $A_1, A_2, \ldots, A_n$ such that $\forall i \in [1..n]\ A_i = \{x_{\sigma(i)}, y_{\rho(i)}\}$ with $x_{\sigma(i)} \in X$, $y_{\rho(i)} \in Y$ and $s(x_{\sigma(i)}) + s(y_{\rho(i)}) = v_i$.

Vice versa, let us assume that we can partition set $X \cup Y$ into n disjoint sets $A_1, A_2, \ldots, A_n$ such that $\forall i \in [1..n]\ A_i = \{x, y\}$ with $x \in X$ and $y \in Y$ and $s(x) + x(y) = v_i$. There is a permutation σ of U and a permutation ρ of D such that $u_{\sigma(i)} + d_{\rho(i)} = v_i$ for $i = 1, 2, \ldots, n$. By joining the columns related to $\sigma(U)$'s elements to $\rho(D)$'s as shown in Fig. 15, we obtain a column-convex polyomino that satisfies (H', V').

Therefore, NMTS $\propto$ RSP on $(\mathbf{p}, \mathbf{v})$ and the theorem is proved. □

By symmetry, the problem RSP on $(\mathbf{p}, \mathbf{h})$ is NP-complete. It can also be proven that NMTS $\propto$ RSP on $(\mathbf{v})$: in the same way, we obtain an instance (H', V') of RSP related to an arbitrary instance of NMTS. If (H', V') is satisfiable on $(\mathbf{v})$, then there is a set Λ with connected columns having its ith row projection and the jth column projection equal to h'_i and v'_j, respectively. However, $h'_k = h'_{k+1} = n$ and so Λ's kth and $(k+1)$th row are n long. Therefore, Λ is a column-convex polyomino. By proceeding in the same way, a partition $A_1, A_2, \ldots, A_n$ of $X \cup Y$ such that $\forall i \in [1..n]\ A_i = \{x_{\sigma(i)}, y_{\rho(i)}\}$ with $x_{\sigma(i)} \in X$, $y_{\rho(i)} \in Y$ and $s(x_{\sigma(i)}) + s(y_{\rho(i)}) = v_i$ can be obtained from Λ.

Vice versa, if a partition of $X \cup Y$ like this exists, then a column-convex polyomino exists that satisfies (H', V'). But $(\mathbf{p}, \mathbf{v}) \subset (\mathbf{v})$ and so (H', V') is satisfiable in $(\mathbf{v})$.

In other words, we have:

Theorem 4.2. *RSP problem on* $(\mathbf{p}, \mathbf{h})$, $(\mathbf{v})$ *and* $(\mathbf{h})$ *is* NP*-complete.*

Let us take a variant of NMTS problem:

Instance: Disjoint sets X and Y, each containing n elements, a function $s: X \cup Y \to \mathbb{N}$ that is polynomial in n, and a vector $V = (v_1, v_2, \ldots, v_n) \in \mathbb{N}^n$.

Question: Can set $X \cup Y$ be partitioned into n disjoint sets $A_1, A_2, \ldots, A_n$ such that $\forall i \in [1..n]$ $A_i = \{x_i, y_i\}$, $x_i \in X$, $y_i \in Y$, and $s(x_i) + s(y_i) = v_i$, with $(s(x_1), s(x_2), \ldots, s(x_n))$ and $(s(y_1), s(y_2), \ldots, s(x_n))$ unimodal sequences?

Here, sequence: $(s(x_1), s(x_2), \ldots, s(x_n))$ and $(s(y_1), s(y_2), \ldots, s(x_n))$ are required to be unimodal, that is, there are $k, h \in [1..n]$ such that:

$$s(x_i) \leqslant s(x_{i+1}), \quad 1 \leqslant i \leqslant k-1,$$

$$s(x_i) \geqslant s(x_{i+1}), \quad k \leqslant i \leqslant n-1,$$

$$s(y_i) \leqslant s(y_{i+1}), \quad 1 \leqslant i \leqslant h-1,$$

$$s(y_i) \geqslant s(y_{i+1}), \quad h \leqslant i \leqslant n-1.$$

Given any instance of this problem, we proceed as before and obtain instance (H', V') of RSP in polynomial time. Consequently, (H', V') is satisfiable in $(\mathbf{p}, \mathbf{v}, \mathbf{h})$ iff there is a partitioning $A_1, A_2, \ldots, A_n$ of $X \cup Y$ such that:

$\forall i \in [1..n]$ $A_i = \{x_i, y_i\}$, $x_i \in X$, $y_i \in Y$,

$\forall i \in [1..n]$ $s(x_i) + s(y_i) = v_i$,

$(s(x_1), s(x_2), \ldots, s(x_n))$ and $(s(y_1), s(y_2), \ldots, s(x_n))$ are unimodal sequences.

In fact, if (H', V') is satisfiable in $(\mathbf{p}, \mathbf{v}, \mathbf{h})$, there is a convex polyomino A that satisfies (H', V') from which we can obtain a unimodal permutation ρ of D and a unimodal permutation σ of U such that $\forall i \in [1..n]$ $u_{\sigma(i)} + d_{\rho(i)} = v_i$ (see Fig. 16). Consequently, set $X \cup Y$ can be partitioned into n disjoint sets $A_1, A_2, \ldots, A_n$ such that $\forall i \in [1..n]$ $A_i = \{x_{\sigma(i)}, y_{\rho(i)}\}$, $x_{\sigma(i)} \in X$, $y_{\rho(i)} \in Y$ and $s(x_{\sigma(i)}) + s(y_{\rho(i)}) = v_i$, with $(s(x_{\sigma(1)}), s(x_{\sigma(2)}), \ldots, s(x_{\sigma(n)}))$ and $(s(y_{\rho(1)}), s(y_{\rho(2)}), \ldots, s(y_{\rho(n)}))$ unimodal sequences. Vice versa, it is easy to see that if such a partition of $X \cup Y$ exists, then there is a convex polyomino that satisfies (H', V').

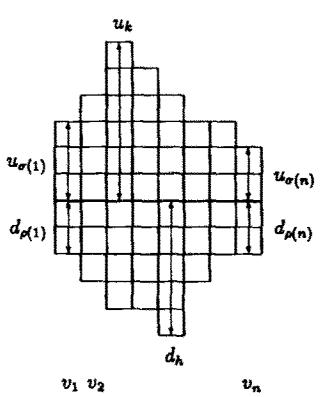

Fig. 16. A convex polyomino that satisfies (H', V').

However, since we have shown that RSP problem on (**p**, **v**, **h**) is solvable in polynomial time, it follows that:

Theorem 4.3. *The NMTS problem is solvable in polynomial time if we require that sequences* $(s(x_1), s(x_2), \ldots, s(x_n))$ *and* $(s(y_1), s(y_2), \ldots, s(x_n))$ *related to partition* $A_1 = (x_1, y_1), A_2 = (x_2, y_2), \ldots, A_n = (x_n, y_n)$ *of* $X \cup Y$ *be unimodal.*

5. Conclusions

In this paper, we examined the problem of determining the existence of a finite discrete set of cells S whose row and column projections are equal to the elements of an assigned pair of vectors (H, V). We studied the problem (denoted RSP) on some classes made up of cell sets satisfying certain convexity conditions. In particular, we showed that *RSP* problem is:

- Solvable in polynomial time on the class of convex polyominoes,
- NP-complete on the class of column-convex polyominoes (row-convex polyominoes),
- NP-complete on the class of sets having connected columns (rows).

Ryser [23] and, subsequently, Chang [5] solved the problem on the class (∅) (that is, when the set S is a general set of cells). Both authors established some algorithms that, starting out with $V \in \mathbb{N}^n$ and $H \in \mathbb{N}^m$ affirm the existence of set S in $O(nm)$ time.

Up to now, the problem on the class (**p**) of polyominoes has not been solved. Only a heuristic algorithm is known for the reconstruction of set S on class (**v**, **h**) of sets having connected rows and columns (see [20]). In short, RSP problem is:

- polynomial on (**p**, **v**, **h**),
- NP-complete on (**p**, **v**),
- NP-complete on (**p**, **h**),
- open on (**v**, **h**),
- NP-complete on (**v**),
- NP-complete on (**h**),
- open on (**p**),
- polynomial on (∅).

References

[1] B. Aspvall, M.F. Plass and R.E. Tarjan, A linear-time algorithm for testing the truth of certain quantified boolean formulas, *Inform. Process. Lett.* **8** (3) (1979) 121–123.

[2] D. Beauquier and M. Nivat, Tiling the plane with one tile, in: *Topology and Category Theory in Computer Science* (Oxford Univ. Press, Oxford, 1991) 291–334.

[3] D. Beauquier, M. Nivat, E. Remila and M. Robson, Tiling figures of the plane with two bars, a horizontal and a vertical one, Rapport LITP92, Université de Paris VII, 1992.

[4] R. Berger, The undecidability of the domino problem, *Mem. Amer. Soc.* **66** (1966).

[5] S.K. Chang, The reconstruction of binary patterns from their projections, *Comm. ACM*, **14** (1971) 21–25.
[6] S.K. Chang and C.K. Chow, The reconstruction of three-dimensional objects from two orthogonal projections and its application to cardiac cineangiography, *IEEE Trans. Comput.* **C-22** (1973) 661–670.
[7] S.K. Chang and G.L. Shelthon, Two algorithms for multiple-view binary pattern reconstruction, *IEEE Trans. Systems Man. Cybernet.* **SMC-1** (1971) 90–94.
[8] S.K. Chang and Y.R. Wang, Three-dimensional objects reconstruction from two orthogonal projections, *Pattern Recognition* 7 (1975) 167–176.
[9] J.H. Conway and J.C. Lagarias, Tiling with polyominoes and combinatorial group theory, *J. Combin. Theory Ser. A* **53** (1990) 183–208.
[10] M. Delest, Polyominoes and animals: Some recent results, *J. Comput. Chem.* **8** (1991) 3–18.
[11] A. Del Lungo, Polyominoes defined by two vectors, *Theoret. Comput. Sci.* **127** (1994) 187–198.
[12] A. Del Lungo, M. Nivat and R. Pinzani, Polyominoes convexes définis par un couple de vecteurs, *Proc. 6th FPSAC*, DIMACS, Piscataway (1994) 79–90.
[13] M. Gardner, Mathematical games, *Sci. Amer.* (1958) September 182–192, November, 136–142.
[14] M.R. Garey and D.S. Johnson, *Computers and Intractability: A Guide to the Theory of* NP-*Completeness* (Freeman, New York 1979) 224.
[15] S.W. Golomb, *Polyominoes* (Scribner, New York, 1965).
[16] S.W. Golomb, Tilings with sets of polyominoes, *J. Combin. Theory* **9** (1970) 60–71.
[17] R. Gordon and G.T. Herman, Reconstruction of pictures from their projections, *Graph. Image Process.* **14** (1971) 759–768.
[18] D.A. Klarner, My life among the polyominoes, in: *The Mathematical Gardner* (Wadsworth, Belmont CA, 1981) 243–262.
[19] S. Krishnan, S.S. Prabhu and E.V. Krishnamurthy, Probabilistic reinforcement algorithms for the reconstruction of pictures from their projections, *Internat. J. Systems. Sci.* **4** (1973) 661–670.
[20] A. Kuba, The reconstruction of two-directionally connected binary patterns from their two projections, *Comput. Vision Graph. Image Process.* **27** (1984) 249–265.
[21] A. Kuba, On the reconstruction of binary matrices from their projections, *Pubbl. dell'Ist di Analisi Globale e Applicazioni*, serie "Problemi ben posti ed inversi", Firenze, 1986.
[22] D.G.W. Omnasch and P.H. Heinstzen, A new approach for the reconstruction of the right or left ventricular form from biplane angiocardiographic recordings, in: *Digital Imaging in Cardiovascular Radiology* (Georg Thieme, Stuttgart, 1983) 151–163.
[23] H. Ryser, *Combinatorial Mathematics*, The Carus Mathematical Monographs, Vol. 14 (The Mathematical Association of America, Rahway, 1963).
[24] A.R. Shliferstein and Y.T. Chien, Switching components and the ambiguity problem in the reconstruction of pictures from their projections, *Pattern Recognition* **10** (1978) 327–340.
[25] C.H. Slump and J.J. Gerbrands, A network flow approach to reconstruction of the left ventricle from two projections, *Comput Graph. Image Process.* **18** (1982) 18–36.
[26] M.M. Ter-Pogossian and M.E. Phelps eds., *Reconstruction tomography in diagnostic radiology and nuclear medicine* (Univ. Park Press, Baltimore, 1977).
[27] X.G. Viennot, Problèmes combinatoires posés par la physique statistique, *Séminaire Bourbaki No. 626, 36ème année*, in: *Astérisque* **121–122** (1985) 225–246.
[28] X.G. Viennot, A survey of polyomino enumeration, Proc. Séries formelles et combinatoires algébrique, Montréal, Juin 1992; *Publications du LACIM* 11, Université du Québec a Montréal, 1992.
[29] Y.R. Wang, Characterization of binary patterns and their projections, *IEEE Trans. Comput.* **C-24** (1975) 1032–1035.
[30] C.K. Wong and C.K. Yue, Reconstruction of patterns by block-projection, *Inform. Sci.* **4** (1972) 357–366.